Experimental Study of a Hybrid Genetic Algorithm for the Multiple Travelling Salesman Problem

Maha Ata Al-Furhud and Zakir Hussain Ahmed

1Department of Computer Science, College of Computer and Information Sciences, Jouf University, Sakakah, Al-Jouf, Saudi Arabia
2Department of Computer Science, College of Computer and Information Sciences, Al Imam Mohammad Ibn Saud Islamic University, Riyadh, Saudi Arabia
3Department of Mathematics and Statistics, College of Science, Al Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia

Correspondence should be addressed to Maha Ata Al-Furhud; m2a8h2a@hotmail.com

Received 19 June 2020; Revised 26 July 2020; Accepted 15 October 2020; Published 27 October 2020

Copyright © 2020 Maha Ata Al-Furhud and Zakir Hussain Ahmed. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The multiple travelling salesman problem (MTSP), an extension of the well-known travelling salesman problem (TSP), is studied here. In MTSP, starting from a depot, multiple salesmen require to visit all cities so that each city is required to be visited only once by one salesman only. It is NP-hard and is more complex than the usual TSP. So, exact optimal solutions can be obtained for smaller sized problem instances only. For large-sized problem instances, it is essential to apply heuristic algorithms, and amongst them, genetic algorithm is identified to be successfully deal with such complex optimization problems. So, we propose a hybrid genetic algorithm (HGA) that uses sequential constructive crossover, a local search approach along with an immigration technique to find high-quality solution to the MTSP. Then our proposed HGA is compared against some state-of-the-art algorithms by solving some TSPLIB symmetric instances of several sizes with various number of salesmen. Our experimental investigation demonstrates that the HGA is one of the best algorithms.

1. Introduction

The travelling salesman problem (TSP) is a well-known problem in computer science and operations research that aims to find a minimum cost Hamiltonian circuit in any network. The problem may be stated as follows: suppose a number of cities (or nodes) along with their interdistances are given. Beginning from and ending at the same depot (or headquarters), a salesman must visit all cities only once. The problem is to find minimum total distance (or cost) toured by the salesman. The TSP is widely studied by numerous researchers, and hence, numerous efficient methods are proposed for solving it. But some problems need extra salesman, and so the multiple salesman problem (MTSP) is introduced that generalizes the TSP. In MTSP, every salesman begins from and ends the journey at the same depot. Except the depot, each city is required to be visited only once by one salesman only. The problem is to find minimum total distance (or cost) toured by the salesmen [1].

The MTSP is identified as more suitable than the usual TSP for some real-world applications. It has an application in the job scheduling that contains various parallel production units [2]. The vehicle scheduling problem (VSP) can be modelled by the problem. The MTSP further models another TSP variant that aims to visit \( n \) cities in \( m \) weeks provided that, during the weekends, the salesman returns to his home city [3]. The problem has application in school bus scheduling that finds a bus filling structure so that the number of tours as well as the distance toured by the busses is minimum provided that no bus is overcrowded and the time needed to cross any way does not surpass the maximum allowable time [4]. Another application of the MTSP is
reported as crew scheduling in [5] where the problem of scheduling numerous groups of photographers to several schools is investigated. Further, print press scheduling [3], interview scheduling [6], mission planning [7], and the global navigation satellite surveying system design [8] are some more applications of the MTSP. There are numerous variations of the MTSP [1] depending on the number of depot cities—single depot or multidepot, and type of path—open or closed. A path is called closed if it starts from and finishes at the same depot city, otherwise, open. This paper considers single depot and closed path MTSP.

The MTSP is NP-hard [9], and a polynomial-time algorithm is not available to solve the problem. Hence, obtaining optimal solution using exact algorithm is very difficult, though it is not impossible. Therefore, for obtaining a good optimal solution, within reasonable computational, to such problems, nowadays, heuristic algorithms are used. Artificial neural network (ANN) [10], simulated annealing (SA) [11], genetic algorithm (GA) [12], particle swarm optimization algorithm (PSO) [13], ant colony optimization (ACO) [14], nearest neighbour method [15], etc., are a few such popular heuristic approaches.

In the past few years, for some difficult optimization problems, like the quadratic assignment problem [16], the minimum spanning tree problem [17], and the TSP [18], several GAs have been successfully developed. GAs are developed based on imitating the “survival of the fittest” theory amongst various species formed by random changes in the structure of chromosomes in natural biology. Since GAs are very flexible, simple, and easy to encode, they are identified to be extremely successful. Beginning with an initial chromosomes’ population, a simple GA passes through generally three operators, namely, selection operator, crossover operator, and mutation operator, to create better populations successively in the following generations. Selection operator copies some chromosomes probabilistically to the subsequent generation. Crossover mates two randomly selected parent chromosomes to form offspring chromosomes. Mutation selects a gene randomly at a chromosome position and alters its value. The crossover combined with selection is the most dominant process in GA search. Mutation operator diverges the GA search space and protects loss of genetic material. Hence, mutation probability is kept very small, whereas crossover probability is kept very high [19].

Though several GAs are proposed for the problem, still research is going on for developing better GA. Crossover operator is the most significant operator in GAs. Generally, crossover operators which were proposed for the usual TSP are applied to the MTSP also. However, majority of these operators cannot obtain good GAs for the problem. Using a good crossover method can obtain valuable GA. An experimental investigation amongst six crossover operators demonstrated that sequential constructive crossover (SCX) is one of the best crossovers [20].

A GA using three basic operators is called simple GA. Though simple GAs can solve combinatorial optimization problems, occasionally they lead to premature convergence. They may get stuck in local minima and take long time to find optimal solutions [16]. So one must apply some advanced approaches to improve the solution quality found by simple GAs. As an advanced approach, generally simple GAs are hybridized by incorporating local search method and immigration method.

In this study, by introducing required dummy depots, the MTSP is first reduced to the usual TSP. Then a hybrid GA (HGA) is developed using a heuristic method for creating initial population, SCX, swap mutation, local search approach, and an immigration approach for finding high-quality solution to the reduced problem. The usefulness of our proposed HGA is observed against some ACO-based algorithms [21–23], GAs [24, 25], and a gravitational emulation approach [26] on some symmetric instances from TSPLIB [27]. The computational experiences demonstrate the usefulness of the proposed HGA. Finally, we report solutions to some additional symmetric instances from TSPLIB.

We organize the paper as follows. Section 2 reviews the associated previous literature studies. In Section 3, the problem definition and formulation are discussed. The hybrid genetic algorithm for the problem is designed in Section 4. Section 5 displays the computational experiences while Section 6 reports conclusions as well as discussions on the finding of this investigation.

2. Literature Review

The MTSP is a very complex NP-hard problem. Methods to solve this problem as well as any other complex optimization problem are classified into two broad categories: exact and heuristic. There are very few literature studies on the MTSP. The first exact approach developed by relaxing some MTSP constraints was proposed in [28] to solve it straightforward without any reduction to the single salesman problem. Exact algorithms based on branch-and-bound approach have been reported in [29]. As this investigation proposes a heuristic method for finding solution to the problem, an extensive literature review on heuristic algorithms is presented. Heuristic approaches are mostly of three categories: constructive, improvement, and limited enumeration approaches. The very recent approaches are called metaheuristics which have been used in numerous optimization problems.

The first heuristics algorithm to the MTSP was developed on the reduced problem to the single salesman on an enlarged graph [30]. The approach is an improved Lin–Kernighan approach which was basically proposed for the TSP [31].

The first GAs for solving the MTSP was proposed in [5]. In [24], GAs are proposed for finding solution to the problem that represents hot rolling scheduling in a steel plant in Shanghai. First, the problem is represented as an MTSP, which is then reduced to a single salesman problem, and finally, a modified GA is applied to find the solution. Several evolutionary algorithms are proposed to solve the MTSP and then compared in [32].

A GA is developed in [3] which uses a new set of chromosomes and associated operators for solving the
MTSP. They investigated theoretical properties and reported the computational performance of their algorithm. The computational result shows that this technique finds better results in lesser search space.

Grouping genetic algorithm-based approach for the MTSP has been proposed which is found to be good [33]. Further, an objective which finds minimum of the maximum distance toured by any salesman is also considered.

A hybrid metaheuristic approach, called GA2OPT, is developed in [34] to find solution of the MTSP. At first, a modified GA is applied to solve the MTSP by allowing various iterations and then 2-opt local search approach is applied for improving solutions in every iteration. The algorithm was applied on six TSPLIB instances and obtained good quality solutions.

A modified ACO, named NMACO that combines swap, 2-opt, and insert algorithms along with an approach to come out from local optimal points, is proposed in [23] for finding solution to the problem. The algorithm is compared against other well-known metaheuristic approaches for some instances. As reported, the proposed NMACO is competitive with other existing metaheuristic approaches.

The two-part chromosome crossover (TCX) is proposed for finding solution to the problem that reduces the size of the search space for the problem [12]. A comparative study is carried out with three different crossover approaches for the biobjective MTSP which aims to minimize total distance toured and the longest tour. As mentioned, TCX obtains better solution than the other crossover operators.

In [35], an evolutionary algorithm that combines an imperialist competitive approach with Lin–Kernighan algorithm is proposed for finding solution to the MTSP. The algorithm also uses an absorption method and some local search methods. The algorithm was tested on 26 instances and found better good results than by another metaheuristic algorithm.

A modified TCX is suggested in [36] for the MTSP. As reported, the algorithm assigns different number of cities for different salesmen and obtains good solutions.

A modified gravitational emulation local search (M-GELS) approach is developed in [26] to find solution to the symmetric MTSP. First, a set of feasible solutions is created using sweep algorithm which is then improved using M-GELS. Computational results show that M-GELS is better than some well-known optimization approaches to solve the problem.

The two-phase heuristic approach that uses an improved K-means approach by collecting the cities depending on their positions and precise capacity constraints is developed for the MTSP [37]. Next, a genetic-based path planning approach is developed to find perfect path for the above set. As reported, compared to the path planning approach purely based on GA, the algorithm shows better performance to solve the problem.

A novel hybrid approach, named AC2OptGA, by combining a modified ACO, 2-opt, and GA is proposed in [21] to find solution to the problem. Initially, solutions are generated using ACO, then enhanced the obtained solutions using the 2-opt method, and finally, once again improved solution quality using GA. This approach is evaluated using various instances from TSPLIB and found better than the existing algorithm.

3. Problem Definition and Its Complexity

The MTSP is a very difficult problem in operations research as well as in computer science that has various real-life applications. The problem may be stated as follows: Suppose \( n \) cities \((1, 2, 3, \ldots, n)\) along with their interdistances, \( d_{ij} \) \((i, j = 1, 2, \ldots, n)\), and \( m \) salesmen are given in a network. Beginning from and ending at the same depot, all salesmen must visit all cities so that each city is required to be visited only once by one salesman only. The problem is to find the optimal order of cities, i.e., tour, by every salesman so that the total distance (or cost) toured by all salesmen is minimized. The problem reduces to the usual TSP when \( m = 1 \).

The MTSP can be defined as an integer linear programming as follows [29]:

\[
\text{minimize } z = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij}x_{ij}, \quad d_{ij} = \infty \text{ for } i = j, \\
\text{subject to, } \sum_{j=2}^{n} x_{ij} = m, \\
\sum_{i=2}^{n} x_{ij} = m, \\
\sum_{i=1}^{n} x_{ij} = 1, \quad j = 2, \ldots, n, \\
\sum_{j=1}^{n} x_{ij} = 1, \quad i = 2, \ldots, n, \\
+\text{subtour elimination constraints,} \\
x_{ij} \in \{0,1\}, \quad \forall (i, j) \in A.
\]

Let \( x_{ij} \) be a binary variable that is equal to 1 (one), if and only if the link \((i, j)\) is present in the tour, otherwise 0 (zero). The total distance traveled by all \( m \) salesmen is represented by the objective function, \( z \), which is to be minimized. The equalities in (1) and (2) confirm that only \( m \) salesmen exit from and come back to “city 1” (depot), whereas (3), (4), and (6) represent assignment constraints. The constraints in (5) prevent subtours that are called subtour removal constraints.

The distances may denote costs, times, etc. Based on the distances, TSPs are classified into symmetric and asymmetric TSPs. It is symmetric when \( d_{ij} = d_{ji} \), \( \forall i, j \), else, asymmetric. There are likely \((n-1)!\) number of tours for \( n \)-city usual TSP. So, the computational effort is proportional to the size of the problem. It is very hard to find solution to larger sized instances, although it is not impracticable. Further, in MTSP, it is required to first decide the cities assigned optimally to every salesman and then to find the optimal order of cities in every salesman’s tour. Hence, the
MTSP is harder than the TSP and it is proved to be NP-hard [9]. There is no any polynomial-time approach for the MTSP.

3.1. Modified Distance Matrix. The problem may be reduced to the usual TSP by considering single salesman. Also, the problem having \( n \) cities with \( m \) salesmen could be reduced to the single salesman having \( n + m - 1 \) cities after introducing \( m - 1 \) dummy depots (specifically, \( n + 1, \ldots, n + m + 1 \)) [38]. It can be also considered as a reduction of the famous vehicle routing problem (VRP) by eliminating capacity constraints [39]. We are going to transfer the MTSP to the usual TSP by introducing \( m - 1 \) dummy depots. One example of the problem instance with \( n = 9, \ m = 2 \) is given in Figure 1(a), while the transformation of this example to the single salesman is shown in Figure 1(b). A randomly generated distance matrix and the modified distance matrix having one dummy depot city 10 for a 9-city with 2-salesman instance are shown in Tables 1 and 2 respectively.

3.2. Alphabet Table. Alphabet matrix, \( A = [a(i,j)] \), is an \( n \times n \) square matrix created by locations of elements of the \( n \times n \) modified distance matrix, \( D' = [d'_{ij}] \), after they are ordered in nondecreasing sequence according to their distances. Alphabet table "\([a(i, j) - d'_{i,\text{all}(j)}]\)" is the mixture of elements (cities) of the matrix \( A \) and their distances in the modified distance matrix [40]. For the modified matrix in Table 2, the alphabet table is shown in Table 3, where \( C \) is a city and \( D \) is its distance from the corresponding city in the first column.

4. A Hybrid Genetic Algorithm for the MTSP

Genetic algorithms (GAs) are identified to be successful for the usual TSP as well as its different variations. However, they do not guarantee that the obtained best solutions are exact optimal; nevertheless, they generally find solutions that are very close to the optimal solutions quickly. We are applying a heuristic approach for creating initial population, then sequential constructive crossover, swap mutation operator, and a local search approach along with an immigration method for developing a hybrid genetic algorithm for the MTSP.

4.1. Chromosome Representation Schemes. To apply GAs to any optimization problem, a chromosome representation method must be found/determined for representing solutions such that genetic operators produce legal chromosomes. For representing solutions of the MTSP, mainly three methods are used, which are one chromosome [24], two chromosomes [41], and two-part chromosome [42]. In addition, modified chromosome representations are found in [33, 43]. We are using one chromosome having dummy depots. An example of the proposed chromosome representation for \( n = 9 \) with \( m = 2 \) is shown in Figure 2.

In the chromosome: \((1, 9, 6, 2, 10, 7, 4, 3, 8, 5)\), there are \( 10 (= 9 + 2 - 1) \) genes including dummy depot city 10. It denotes the tour \( \{1 \rightarrow 9 \rightarrow 6 \rightarrow 2 \rightarrow 10 \rightarrow 7 \rightarrow 4 \rightarrow 3 \rightarrow 8 \rightarrow 5 \rightarrow 1\} \). Here, the cities 1, 9, 6, and 2 are visited in the order by \( 1^{\text{st}} \) salesman, and the cities 7, 4, 3, 8, and 5 are visited in the order by \( 2^{\text{nd}} \) salesman. For this tour, the total distance toured by all salesmen, \( (10 + 7 + 13 + 11 + 8 + 7 + 11 + 13 + 3 + 8) = 91 \), is the objective function value.

4.2. Initial Population by Sequential Sampling. Beginning with a better initial population can give better solution quality quickly, and so, several GAs used heuristic approaches to create initial population. For this, we are going to use a probabilistic and simple form of sequential constructive sampling approach [44]. It was used successfully for other variants of the TSP [45-47].

We first construct an alphabet table on the modified distance matrix and then assign probability of travelling each untravelled city in any row so that the probability for the first untravelled city is assigned more than that of the \( 2^{\text{nd}} \) one, and so on. Then cumulative probability is calculated for each unvisited city in that row. After that, any random number, \( r \in [0, 1] \), is produced and the city representing the random number in any cumulative probability interval is considered. The probability of accepting any unvisited city is calculated...
as follows. Let the number of available (unvisited) cities be \( k \) in a row of the table. The probability of accepting the \( i^{th} \) available city is \( p_i = \frac{2(k-i+1)}{(k+1)} \). The sequential sampling algorithm [48] is reported in Algorithm 1.

The algorithm is illustrated using the example shown in Table 1 with 9 cities and 2 salesmen. In 1st row, the number of available (unvisited) cities is 8, which are cities 6, 2, 7, 4, 8, 5, 9, and 3, along with their cumulative probabilities 0.22, 0.42, 0.58, 0.72, 0.83, 0.92, 0.97, and 1.00, respectively. Let the generated random number be 0.04. Since 0.04 is less than cumulative probability 0.22 for the first city in the sequence, so the first city, which is 6, is selected and the partial chromosome becomes (1, 6). Then go to 6th row of the table. The number of available cities in this row is 7, which are 8, 5, 7, 9, 10, 3, and 4, with cumulative probabilities 0.22, 0.42, 0.58, 0.72, 0.83, 0.92, 0.97 and 1.00 respectively. Let the generated random number be 0.47, then city 7 is selected and the partial chromosome becomes (1, 6, 7). Then go to 7th row of the table, where the number of available cities in this row is 7, which are 3, 10, 4, 8, 9, 5, and 2, with cumulative probabilities 0.25, 0.46, 0.64, 0.79, 0.89, 0.96, and 1.00, respectively. Let the generated random number be 0.33, then city 10 is selected and the partial chromosome becomes (1, 6, 7, 10). In this 10th row, the number of available cities is 6 that are 8, 5, 7, 9, 10, and 3, along with their cumulative probabilities 0.29, 0.52, 0.71, 0.86, 0.95, and 1.00, respectively. Let the generated random number be 0.50, then city 4 is selected and the partial chromosome becomes (1, 6, 7, 10, 4). Continuing in this manner, one can have a chromosome (1, 6, 7, 10, 4, 9, 8, 2, 3, 5).

A primary study demonstrates the usefulness of this approach for creating initial population. However, as observed, for selecting a city, in place of choosing all available cities if only a maximum of the first ten cities in a row is considered, then this approach creates better initial population ([45-47]). Hence, for our study also, this limited domain of available cities is considered. In addition, 2-opt search is applied to each chromosome for improving the population.

4.3. Fitness Function and Selection Operator. Since this problem is a minimization problem, the objective and fitness functions are different. One can define a fitness function as \( F(x) = 1/(1+f(x)) \), where \( f(x) \) is objective function. For example, if \( f(x) \) is objective function cost, then its fitness function cost is \( 1/92 = 0.0109 \) (approximately). A shorter tour (chromosome) will have a higher fitness function value that will also have a higher probability to select in the next mating pool. In selection operation, no any new chromosome is built. We considered the following selection method. At first, the expected count of every chromosome in a population is calculated by dividing the corresponding fitness function value by the average fitness function value. Next each chromosome is copied as many times as the mantissa value (integer part value) of its expected count, and each mantissa is subtracted from the corresponding expected count. This will lead the value of all expected counts less than one (1). Then, a number \( r \) and a chromosome are randomly selected. Then the chromosome is inserted into its mating pool if \( r \) is less than its expected count and 0.5 is subtracted from its expected count. Repeat this procedure until the mating pool full. This method is called stochastic remainder selection process [49] that is reported in Algorithm 2.

4.4. Sequential Constructive Crossover Operator. In GAs, the crossover operator plays an important role. Ordered crossover (OX) [50], cycle crossover (CX) [51], partially mapped crossover (PMX) [52], edge recombination crossover (ERX) [53], alternating edge crossover (AEX) [53], and sequential constructive crossover (SCX) [54] are some of the broadly applied crossovers for the usual TSP which are also applied to the MTSP. However, no one can apply these operators directly with the two-part chromosome method. The crossover OX is applied with a crossover, denoted as OX + A, which was reported for two-part chromosome in [3]. In [12], a two-part chromosome crossover (TCX) is proposed for the problem and then compared with PMX + A, OX + A, and CX + A, and found that TCX is better. Al-Omeer and Ahmed [20] used one chromosome representation and compared crossovers—PMX, OX, ERX, AEX, CX, and SCX for the MTSP, and found that SCX is the best one. SCX sequentially examines the pair of parent chromosomes for selecting 1st legitimate (untravelled) cities appeared after the current city. If no legitimate city exists in

<table>
<thead>
<tr>
<th>City</th>
<th>C-D</th>
<th>C-D</th>
<th>C-D</th>
<th>C-D</th>
<th>C-D</th>
<th>C-D</th>
<th>C-D</th>
<th>C-D</th>
<th>C-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6-6</td>
<td>2-7</td>
<td>7-8</td>
<td>4-9</td>
<td>8-9</td>
<td>5-10</td>
<td>9-10</td>
<td>3-15</td>
<td>1-999</td>
</tr>
<tr>
<td>2</td>
<td>6-3</td>
<td>9-3</td>
<td>8-4</td>
<td>7-6</td>
<td>4-7</td>
<td>3-8</td>
<td>1-11</td>
<td>5-11</td>
<td>10-11</td>
</tr>
<tr>
<td>3</td>
<td>9-4</td>
<td>2-5</td>
<td>6-5</td>
<td>7-8</td>
<td>5-12</td>
<td>8-13</td>
<td>1-15</td>
<td>10-15</td>
<td>4-16</td>
</tr>
<tr>
<td>4</td>
<td>1-2</td>
<td>9-2</td>
<td>10-2</td>
<td>8-4</td>
<td>5-2</td>
<td>5-9</td>
<td>3-11</td>
<td>6-13</td>
<td>7-14</td>
</tr>
<tr>
<td>5</td>
<td>3-3</td>
<td>4-5</td>
<td>2-6</td>
<td>6-6</td>
<td>7-7</td>
<td>1-8</td>
<td>10-8</td>
<td>9-9</td>
<td>8-10</td>
</tr>
<tr>
<td>6</td>
<td>8-4</td>
<td>5-5</td>
<td>7-5</td>
<td>9-5</td>
<td>1-6</td>
<td>10-6</td>
<td>3-8</td>
<td>4-11</td>
<td>2-13</td>
</tr>
<tr>
<td>7</td>
<td>3-3</td>
<td>1-5</td>
<td>10-5</td>
<td>6-6</td>
<td>4-7</td>
<td>8-8</td>
<td>9-9</td>
<td>5-12</td>
<td>2-15</td>
</tr>
<tr>
<td>8</td>
<td>2-3</td>
<td>5-3</td>
<td>7-8</td>
<td>1-9</td>
<td>3-9</td>
<td>10-9</td>
<td>9-10</td>
<td>6-11</td>
<td>4-14</td>
</tr>
<tr>
<td>9</td>
<td>3-3</td>
<td>6-7</td>
<td>4-9</td>
<td>7-9</td>
<td>5-10</td>
<td>8-10</td>
<td>1-11</td>
<td>10-11</td>
<td>2-16</td>
</tr>
<tr>
<td>10</td>
<td>6-6</td>
<td>2-7</td>
<td>7-8</td>
<td>4-9</td>
<td>8-9</td>
<td>5-10</td>
<td>9-10</td>
<td>3-15</td>
<td>1-999</td>
</tr>
</tbody>
</table>

Figure 2: An example of our one chromosome for 9 cities with 2 salesmen.
the parent, it examines from the 1st gene of the parent. Once two cities from the parents are selected, their distances from the current city are compared, and the better city is added to the offspring. However, if any infeasible offspring chromosome is created, then it exchanges selected cities to create a feasible chromosome. Al-Furhud and Ahmed [55] developed simple GAs using the original SCX and some modified SCX operators, namely, adaptive SCX [56], greedy SCX [57], reverse greedy SCX [58], and comprehensive SCX [58] for solving the problem, and found that the comprehensive SCX is the best one. However, in this study, we consider the original SCX as the crossover operator in our proposed hybrid GA. Algorithm 3 reports the algorithm for the SCX [54, 55].

Let us demonstrate SCX using the following pair of chromosomes having 9 cities and 2 salesmen, \( P_1: (1, 3, 5, 10, 7, 9, 6, 2, 4, 8) \) and \( P_2: (1, 2, 3, 4, 10, 6, 9, 7, 8, 5) \) having total distances 92 and 72, respectively, on the modified distance matrix (Table 2).

The 1st gene is city 1, after which cities 3 and 2 are respective legitimate (legal) cities in the parents \( P_1 \) and \( P_2 \) with \( c_{13} = 15 \) and \( c_{12} = 7 \). As \( c_{12} < c_{13} \), city 2 is accepted. So, the offspring becomes \( (1, 2) \). After city 2, cities 4 and 3 are the legal cities in \( P_1 \) and \( P_2 \), respectively, with \( c_{24} = 7 \) and \( c_{23} = 8 \). Since \( c_{24} > c_{23} \), we accept city 4. So, the incomplete offspring becomes \( (1, 2, 4) \). After the current city 4, cities 8 and 10 are the legal cities in \( P_1 \) and \( P_2 \), respectively, with \( c_{48} = 4 \) and \( c_{4, 10} = 2 \). Since \( c_{4, 10} < c_{48} \), we accept city 10. So, the incomplete offspring becomes \( (1, 2, 4, 10) \). The legitimate city after city 10 in \( P_1 \) is 7 and in \( P_2 \) is 6 with \( c_{10, 7} = 8 \) and \( c_{10, 6} = 6 \). Since \( c_{10, 6} < c_{10, 7} \), we accept city 6. So, the incomplete offspring becomes \( (1, 2, 4, 10, 6) \). Cities 8 and 9 are legitimate cities after city 6 in \( P_1 \) and \( P_2 \), respectively, with \( c_{68} = 4 \) and \( c_{69} = 5 \). Since \( c_{68} < c_{69} \), we accept city 6. So, the incomplete offspring becomes \( (1, 2, 4, 10, 6, 8) \).

**Algorithm 1: Sequential sampling algorithm.**

**Input:** number of cities \( n \), population size \( P_s \), alphabet table.

**Output:** population of chromosomes.

**for** \( i = 1 \) to \( P_s \) **do**

Set \( p = 1 \).

Current chromosome contains only “city 1.”

**for** \( j = 2 \) to \( n \) **do**

Go to the \( p^{th} \) row of the "Alphabet table."

Collect “legitimate cities” (unvisited cities) sequentially from that row.

Calculate probabilities and cumulative probabilities.

Select a city (suppose city \( q \)) probabilistically and add to the current chromosome.

Rename the “city \( q \)” as “city \( p \)” and continue.

**end for**

**end for**

Return the population

**Algorithm 2: Stochastic remainder selection algorithm.**

**Input:** population size \( P_s \), population of chromosomes.

**Output:** new population of chromosomes.

Calculate the fitness of each chromosome \( f_i \), \( (1 \leq i \leq P_s) \), and average fitness of the population, \( F \).

Initialize number of chromosomes in new population \( j = 0 \).

**for** \( i = 1 \) to \( P_s \) **do**

Calculate expected count \( E_i = f_i / F \).

Set \( m = \) integer part of \( E_i \) and calculate \( E_i - m \).

if \( (m \geq 1) \) then

Copy the chromosome \( i \) exactly equal to \( m \) to the population and update \( j = j + m \).

**end if**

**end for**

while \( j \leq P_s \) do

**for** \( i = 1 \) to \( P_s \) **do**

Generate random number \( r \in [0, 1] \).

if \( (r \leq E_i) \) then

Copy the chromosome \( i \) to the population and update \( j = j + m \).

Calculate \( E_i = E_i - 0.5 \).

**end if**

**end for**

**end while**

Return the new population
becomes (1, 2, 4, 10, 6, 8). There is no legitimate city after city 8 in \( P_1 \), and in \( P_2 \), legitimate city after city 8 is 5 with \( \varepsilon_{85} = 3 \). So, for \( P_1 \), we examine from the 1st gene of the chromosome and select the legitimate city 3 with \( \varepsilon_{35} = 9 \). Since \( \varepsilon_{85} < \varepsilon_{35} \), we accept city 5. So, the incomplete offspring becomes (1, 2, 4, 10, 6, 8, 5). Following this way, a complete offspring (1, 2, 4, 10, 6, 8, 5, 3, 9, 7) with distance 50 is created. Figure 3 displays parents and offspring chromosomes. Generally, a crossover operator that conserves better features of parents in their offspring is said to be a good operator, and SCX is supposed to be good enough in this respect. In Figure 3(c), bold six edges exist in any parent.

As reported in [54], this SCX gets trapped in local optima very quickly because of the identical population. To overcome such issue, selected two parents for mating are examined for replication. If they are identical, the 2nd parent is updated temporarily by exchanging some pair of genes randomly, and then, only the SCX is applied. To keep a combination of parent and offspring in a population, the 1st parent is substituted by the created offspring if the offspring is better than the parent. Now, since the SCX creates single offspring, so, to maintain the same population size in the next generations, when mating with the next chromosome, the present 2nd parent will be treated as the 1st parent, and so on.

4.5. Swap Mutation Operator. The basic mutation operators usually select a gene (or position) randomly in a chromosome and then change its corresponding allele (city). Since incompetent chromosomes are excluded in selection as well as in crossover operators in previous generations, so, some stronger chromosomes’ structures might be lost forever. So, mutation operator is generally applied to recover them. Also, mutation can support other genetic operators to defeat local optima issue and thus find better solution. We have applied swap mutation [19] that selects a pair of cities randomly, excluding dummy depots, and swaps them. The swap mutation approach is presented in Algorithm 4.

In GAs, mutation operator plays a secondary role, and so, the mutation probability is chosen to be significantly less than the crossover probability. Suppose chromosome (1, 2, 4, 10, 6, 8, 5, 3, 9, 7) with distance 50 be chosen for mutation operation, and cities 4 and 7 are swapped, then the chromosome after mutation becomes (1, 2, 7, 10, 6, 8, 5, 3, 9, 4) with distance 49 which are depicted in Figure 4. In Figure 4(b), bold edges are new edges in the chromosome after mutation.

4.6. A Local Search. Numerous local search approaches present in the literature, among them combined mutation is identified as a very good local search approach. It is initially developed for the bottleneck TSP [45] and then applied to maximum TSP [46] and clustered TSP [47] successfully. It merges swap, inversion, and insertion mutations with 1.00 probabilities. In insertion mutation, a city (gene) in a chromosome is selected and then inserted into its random position. In inversion mutation, two points in a chromosome are selected and the subchromosome between them is inverted. For our GA also, this combined mutation is chosen as the local search method. Suppose \( (1 = \alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_n) \) be a chromosome, then the combined mutation [48] for this problem is reported in Algorithm 5.

4.7. Immigration. Due to identical population, very often GAs may get trapped in local optima for solving many complex optimization problems. To overcome such situation, the population must be diversified. Apart from the mutation, another method to diversify the population is the use of immigration method, which randomly replaces some chromosomes by new chromosomes [47]. We use an

---

**Algorithm 3:** Sequential constructive crossover algorithm.

**Input:** crossover probability \( P_c \), pair of chromosomes, cost matrix \( C = [c_{ij}] \).

**Output:** offspring chromosome.

Generate random number \( r \in [0, 1] \).

if \( (r \leq P_c) \) then

Set \( P = 1 \).

The offspring chromosome contains only “city 1.”

for \( i = 2 \) to \( n \) do

In each chromosome consider the first “legitimate city” appeared after “city \( p \).”

if “legitimate city” is not available in a chromosome, then

Search from its 1st gene and select the first “legitimate city” appeared after “city \( p \).”

end if

Suppose “city \( \alpha \)” and “city \( \beta \)” are selected in 1st and 2nd chromosomes, respectively.

if (\( c_{\alpha \alpha} < c_{\beta \beta} \)) then

Add “city \( \alpha \)” to the offspring chromosome.

else

Add “city \( \beta \)” to the offspring chromosome.

end if

Rename the present city as “city \( p \)” and continue.

end for

end if

Return the offspring chromosome.
immigration technique that replaces 25% of the current population arbitrarily using the sampling approach discussed in Section 4.2, if no improvement of the current best solution is seen within the last 10 generations. After applying the immigration, continue with the new population for the next 10 generations for an improvement. The above immigration method and the conditions are fixed after many experiments.

Hence, for the MTSP, our proposed hybrid GA (HGA) is presented in Algorithm 6.
5. Computational Experience

Our proposed HGA is encoded in Visual C++. To demonstrate the effectiveness of our algorithm, a computational experiment is conducted on some standard TSPLIB instances [27] of numerous sizes and then executed on a Laptop with i3-3217U CPU@1.80 GHz and 4 GB RAM under MS Windows 7. We have considered two small-sized problem instances, pr76 and pr152; two medium-sized instances, pr226 and pr299; and two large-sized instances, pr439 and pr1002. For ensuring equally distributed workload among all salesmen, one parameter, \( u \), is initiated that entitles the capacity of every salesman. The capacity is the maximum number of cities that can be toured by a salesman. Our proposed HGA is executed with \( m = 5 \) for the instances.

GA success depends on good selection of some parameters—termination criterion, population size, mutation probability, and crossover probability. But there is no smart way to choose these parameters. One can choose them by trial and error process. Our HGA is executed for numerous parameter sets, and the parameters that are reported in Table 4 are selected.

We compare our proposed HGA with some state-of-the-art algorithms on the abovementioned six instances using \( m = 5 \). The state-of-the-art algorithms are a hybrid approach that combines ACO, 2-opt, and GA algorithms (AC2OptGA) [21], GA with local operators (GAL) [25], modified sweep and ant colony algorithm (SW + AS\textsubscript{elite}) [22], modified gravitational emulation local search (M-GELS) [26], modified GA (MGA) [24], and novel modified ACO (NMACO) [23]. We used the same parameters for our HGA which are used for all of these algorithms.

We report distances of best solution (BS) and average solution (AS) for each instance in Table 5. However, information is no available (NA) for the best solutions by M-GELS. Table 6 reports the percentage of improvement of HGA against the existing six algorithms. In the table, the better working of HGA is indicated by a negative number.

In terms of best and average solutions, Tables 5 and 6 show that our proposed HGA obtains better solutions than all other six algorithms considered here for all sized instances, except for the instance Pr76, GAL and SW + AS\textsubscript{elite} obtain better best solution whereas M-GELS obtains better average solution. The percentage of differences between average solution by HGA and other six algorithms is shown in Figure 5. Tables 5 and 6 and Figure 5 show that the proposed HGA is the best amongst these seven algorithms.

The percentage of difference between the best and average solutions obtained by HGA, AC2OptGA, GAL,
The percentage of differences
AC2OptGA GAL SW + ASelite M-GELS MGA NMACO
pr226 226 50 147586 150950 161084 163235 148050 156542 167665 168156 NA 160350 166827 178501 167239 167821
pr152 152 40 114752 116136 127520 132057 115873 128053 127791 128004 NA 119205 127839 133337 127781 127988
pr76 76 20 154654 157222 159289 163120 153389 162810 157495 157562 NA 147734 157444 160574 157413 157499

Table 7: Comparative study of HGA, AC2OptGA, GAL, SW + ASelite, M-GELS, MGA, and NMACO.

<table>
<thead>
<tr>
<th>Instance</th>
<th>AC2OptGA</th>
<th>GAL</th>
<th>SW + ASelite</th>
<th>M-GELS</th>
<th>MGA</th>
<th>NMACO</th>
</tr>
</thead>
<tbody>
<tr>
<td>pr76</td>
<td>–3.00</td>
<td>–3.75</td>
<td>–0.82</td>
<td>–3.55</td>
<td>NA</td>
<td>6.03</td>
</tr>
<tr>
<td>pr152</td>
<td>–11.13</td>
<td>–13.71</td>
<td>–0.98</td>
<td>–10.26</td>
<td>NA</td>
<td>–2.64</td>
</tr>
<tr>
<td>pr226</td>
<td>–9.15</td>
<td>–8.14</td>
<td>–0.31</td>
<td>–3.70</td>
<td>NA</td>
<td>–6.23</td>
</tr>
<tr>
<td>pr299</td>
<td>–7.08</td>
<td>–9.27</td>
<td>–0.39</td>
<td>–5.50</td>
<td>NA</td>
<td>–4.37</td>
</tr>
<tr>
<td>pr439</td>
<td>–8.05</td>
<td>–8.24</td>
<td>–3.80</td>
<td>–4.36</td>
<td>NA</td>
<td>–9.88</td>
</tr>
<tr>
<td>pr1002</td>
<td>–5.99</td>
<td>–4.43</td>
<td>–0.86</td>
<td>–0.18</td>
<td>NA</td>
<td>–4.60</td>
</tr>
</tbody>
</table>

Table 6: The percentage of differences between HGA and AC2OptGA, GAL, SW + ASelite, M-GELS, MGA, and NMACO.

Figure 5: The percentage of difference between HGA and other algorithms for six instances.
Conclusions and Discussions

A hybrid genetic algorithm (HGA) is presented here to find better solutions to the MTSP. In our algorithm, a sampling approach for creating initial population, sequential constructive crossover, swap mutation operator, and a local search approach along with an immigration technique are used in our proposed HGA. The effectiveness of the HGA to the MTSP against six state-of-the-art algorithms, namely, AC2OptGA, GAL, SW + AS\text{elite}, MGA, NMACO, and M-GELS, is examined for the six symmetric TSPLIB instances with five salesmen. Our comparative study demonstrates that our HGA is the best algorithm amongst seven algorithms considered here.

In addition, to decide if HGA average solution is significantly away from the average solutions found by other six algorithms, we performed a two-tailed Wilcoxon signed-rank test for paired samples and found a significant difference between obtained solutions. We further reported a comparative study between the proposed HGA and AC2OptGA for a total of eighteen benchmark instances of various sizes and various number of salesmen. It is concluded and confirmed by the Wilcoxon test that HGA is more suitable for handling various datasets.

Though the proposed HGA obtains a wide variety of very effective solutions with minor differences between best and average solutions, we agree that yet there is a possibility for improvement of the solutions by incorporating better local search and immigration methods to the problem instances, which is our future investigation.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.
Acknowledgments

This research was funded by King Abdulaziz City for Science and Technology (KACST), Saudi Arabia, vide grant no. 1-17-00-008-0002. The authors are very grateful to the KACST for its financial support.

References


